

Home > Containers and virtualization

NEWS

WebAssembly tech infiltrates the cloud via edge computing

WebAssembly's first beachhead in its mission to break out beyond the web browser has been in edge computing, where it offers efficient support for bespoke serverless workloads.

Beth Pariseau, Senior News Writer

Published: 07 Feb 2022



WebAssembly has moved beyond web browsers and into edge computing as it makes its way toward the heart of cloud and enterprise data centers.

WebAssembly (Wasm), released by the World Wide Web Consortium ([W3C](#)) in 2017, is an assembly language designed to run application code inside web browsers. In 2019, W3C member Mozilla introduced a WebAssembly System Interface (WASI), which created a framework to let WebAssembly apps access operating system resources, setting the stage for WebAssembly to break out of the browser.

ComputerWeekly.com

E - G U I D E

**11 KEY ASPECTS TO
CONSIDER PRIOR TO A
DATA CENTRE UPGRADE**

DOWNLOAD NOW

And break out of the browser it did. Such was the potential for the combination of

WebAssembly and WASI that it prompted Docker founder Solomon Hykes to [tweet](#) in 2019, "If WASM+WASI existed in 2008, we wouldn't have needed to [create] Docker. That's how important it is. WebAssembly on the server is the future of computing."

Since then, WebAssembly has worked its way into [cloud-native computing](#) circles wherever developers want to run custom applications without direct access to the underlying infrastructure. Such environments include mobile and [edge computing](#) deployments, such as Content Delivery Networks ([CDNs](#)), where [WebAssembly offers a lightweight way](#) to move application processing closer to consumers and supports a range of processor types.

While it's still early days for WebAssembly and WASI, as 2022 begins, these concepts are beginning to garner awareness among mainstream IT pros. Despite their promise, however, WebAssembly and WASI also come with drawbacks: They require a separate set of still-maturing compiler toolchains that convert mainstream programming languages into compatible [bytecode](#) and may raise their own security concerns.



Solomon Hykes / @shykes@hachyderm.io

@solomonstre · Follow




If WASM+WASI existed in 2008, we wouldn't have needed to created Docker. That's how important it is. Webassembly on the server is the future of computing. A standardized system interface was the missing link. Let's hope WASI is up to the task!



Lin Clark @linclark

WebAssembly running outside the web has a huge future. And that future gets one giant leap closer today with...

 Announcing WASI: A system interface for running WebAssembly outside the web (and inside it too)

[hacks.mozilla.org/2019/03/standa...](https://hacks.mozilla.org/2019/03/standards-for-webassembly/)

9:39 PM · Mar 27, 2019



2.2K



Reply



Share

[Read 32 replies](#)

WebAssembly pros and cons

Wasm proponents say it's inherently more secure than traditional application runtimes, citing the fact that WebAssembly code is "[sandboxed](#)," or limited in what memory resources it can access, by default.

"One of the issues with software supply chain security is, I have an [NPM](#) package with 100 dependencies, and I have no idea if those 100 dependencies are sending data back to another server," said Michael Yuan, CEO of Second State, a startup that markets a commercial product based on the Cloud Native Computing Foundation's WasmEdge WebAssembly server-side project. "So, with WebAssembly, you can say, 'This module does not have network access, although it's deployed on a network machine.'"

It feels a little bit like going back to how you built software 15 years ago -- once the toolchain's running, it's fine, but most people are not interested in going through all the hassle of setting that up.

Fintan Ryan



IT
Operations



However, while WebAssembly's sandboxing protects against some forms of attack, it remains open to others that have been effectively mitigated for other codebases.

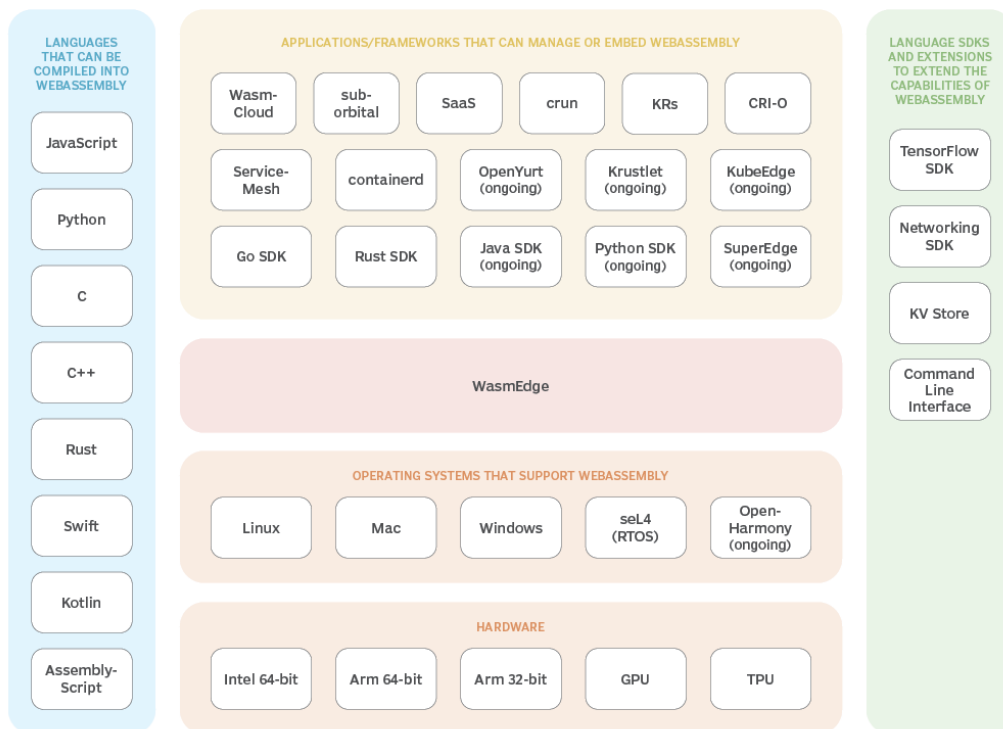
"The sandboxed memory that makes it almost impossible for WebAssembly to touch what is *outside* also makes it harder for the operating system to prevent bad things from happening *inside*," reads a Linux Foundation [blog post](#) from March 2021 (italics in the original). "Traditional memory monitoring mechanisms like 'stack canaries,' which notice if some code tries to mess with objects that it should not touch, cannot work there."

Another potential pitfall for Wasm/WASI is a relative lack of maturity among [compiler](#) tools and toolchains compared to the [DevOps pipelines](#) IT pros have become accustomed to over the last five years, according to Fintan Ryan, an analyst at Gartner.

"It doesn't have the sleekness that a lot of other environments have," Ryan said.

"There's a lot of work happening there, but it feels a little bit like going back to how you built software 15 years ago -- once the toolchain's running, it's fine, but most people are not interested in going through all the hassle of setting that up."

The WebAssembly landscape



SOURCE: MICHAEL YUAN, SECOND STATE

©2022 TECHTARGET, ALL RIGHTS RESERVED 

WebAssembly creeps in around the edges

Given the remaining barriers to early adoption, Wasm has found its way into the corners of the tech industry with the strongest incentives to use it to improve edge scalability and performance.

For example, fast application instantiation using WebAssembly appeals to service providers in the retail industry, such as [Shopify](#), which uses Wasm-based tools to host partners' custom applications within its cloud while maintaining its central platform's security and scalability.

Over the last four years, CDN providers have also begun to support serverless edge computing mechanisms that allow customers to run resource-intensive applications close to consumers, without needing to access or manage the underlying infrastructure.

These mechanisms, in turn, have played host to WebAssembly features outside the traditional web browser domain. Cloudflare, for example, added WebAssembly

support to its [Cloudflare Workers](#) product in 2018. Cloudflare Workers were originally based in JavaScript, but WebAssembly was better suited to a wider range of programming languages and more resource-intensive tasks such as resizing images or processing audio streams, according to a Cloudflare [blog post](#). However, while Cloudflare Workers represent a novel use of Wasm, they use the JavaScript Service Worker API rather than WASI.

Another CDN provider, Fastly, embraced WASI within [Lucet](#), the company's implementation of a WebAssembly compiler and runtime, which Fastly donated to open source in early 2019. By mid-2020, Fastly [shifted its focus](#) from Lucet to Wasmtime, a more broadly supported project governed by the Bytecode Alliance.

This alliance was founded by Mozilla, Fastly, Intel and Red Hat in late 2019 with a focus on improving [Wasm security](#). The alliance also created WASI common, a reusable library of WASI functions, or hostcalls, to further standardize how Wasm projects utilize WASI.

Still, for most enterprise users in late 2021, WebAssembly seemed the domain of specialist service providers such as Fastly, given its relative newness and complexity.

"WebAssembly is super interesting in terms of how it relates to the container space, because there's some question about whether WebAssembly-style sandboxing and runtimes are the future," said Josh Koenig, co-founder and chief strategy officer at Pantheon, a web operations platform in San Francisco.

Pantheon has experimented with Fastly's Compute@Edge platform, which includes WebAssembly, but hasn't yet put it into production. While Koenig has an academic interest in broader WebAssembly discussions, it's unlikely he'll dig any deeper into WebAssembly in his environment, he said.

"It's lower-level than we will probably be involved with as a company," Koenig said. "But we expect that our providers will probably run WebAssembly on our behalf."

Some WebAssembly proponents have ambitions for the technology that go far beyond the edge, deep into enterprise and cloud data centers alongside containers and Kubernetes. Find out how the Cloud Native Computing Foundation is cultivating WebAssembly server-side tools for enterprises in [part two](#) of this story.

Beth Pariseau, senior news writer at TechTarget, is an award-winning veteran of IT journalism. She can be reached at bpariseau@techtarget.com or on Twitter [@PariseauTT](#).

Related Resources

Storage for containers and virtual environments

–TechTarget ComputerWeekly.com

How 5G affects data centres and how to prepare

–TechTarget ComputerWeekly.com

We Do For Storage What Vmware Did For Servers

–StorOne

Dig Deeper on Containers and virtualization

Server-side WebAssembly takes shape, but faces challenges

By: Beth Pariseau

WebAssembly

By: Stephen Bigelow

10 WebAssembly questions to test your Wasm knowledge

By: James Montgomery

Server-side Wasm boosts K8s bonds, devx ahead of key update

By: Beth Pariseau

-ADS BY GOOGLE

[SOFTWARE QUALITY](#) [APPLICATION ARCHITECTURE](#) [CLOUD COMPUTING](#) [AWS](#) [JAVA](#) [DATA CENTER](#)

Software Quality

Is a continuous planning process in DevOps worth it?

Is a continuous planning strategy right for your organization's DevOps efforts? Know the practical benefits and challenges, as ...

Google's DORA DevOps report warns against metrics misuse

This year's DORA DevOps report echoes the experiences of one organization that has applied them in practice: DORA metrics can be ...

[About Us](#) [Editorial Ethics Policy](#) [Meet The Editors](#) [Contact Us](#) [Advertisers](#) [Partner with Us](#) [Media Kit](#)
[Corporate Site](#)

[Contributors](#) [Reprints](#) [Answers](#) [Definitions](#) [E-Products](#) [Events](#) [Features](#)

[Guides](#) [Opinions](#) [Photo Stories](#) [Quizzes](#) [Tips](#) [Tutorials](#) [Videos](#)

All Rights Reserved,
Copyright 2016 - 2023, TechTarget

[Privacy Policy](#)

[Cookie Preferences](#)
[Do Not Sell or Share My Personal Information](#)